1. Introduction

In this ODD, we are going to describe the trade-offs made (e.g., buy vs. build, memory space vs. response time), guidelines and conventions etc..We will also discuss the boundary cases, exception handling mechanisms, definitions and acronyms, the decomposition of subsystems into packages and class interfaces.

1.1 Object Design Trade-Offs

To start off, we chose response time and traded off memory thus, we opted to go with a 2-tier client-server architectural style and a DBSM. By choosing this architectural style we effectively enhance the speed of our system thereby improving response time (which was also part of our non-functional requirements(NF)) because the communication between the server and client is direct. Moving on to the buy vs. build conundrum, this was a challenging task as such, we tried to balance it. We made use of off-the-shelf software as well as making custom software in order to tailor the need for our system. Some of the reasons for choosing off-the-shelf software include time constraint; we have a deadline to submit our proposed system and also a myriad of other works going on, in addition, it would also be unwise to create somewhat of a duplicate seeing as some of the software we need already exists so we can reuse.

For boundary cases, this is used in testing phase and it how the system behaves when a particular input is tested at its maximum, minimum and within range limits. Lets use the password field as an example. Before a user creates a password, there is a slight restriction (for security purposes) making the minimum characters of the password to be at least 6. So in this case, we will take our boundary cases to be 0, 1, 4, 6, 8, 15. This is done in order to test the system and see what behaviour is exhibited when the input restriction(s) of this field is violated. For exception handling mechanism, we opted to go with the common try, catch and throw method to handle exceptions. This is to identify any unexpected occurrences in our program also known as the exceptions; it is an effective method for error handling. The try function

triggers the exception so the exception will be thrown, the code will continue normally if the exception doesn't trigger. The catch retrieves the exception and subsequently creates an object with information about the exception. Our naming conventions were mainly centred around the purpose of the method, class or function we were creating. We also factored some widely used conventional guidelines such as classes generally have singular nouns, methods are names with verbs because methods intend to achieve something in a verb sense (an action) like the vote() method.

1.2 Interface Documentation Guidelines

- System Status: the system informs the user of their actions and shows causality
- Comfort and Freedom in Usage: the system offers the user a chance to retrace their steps or undo their actions
- Exception Handling: errors are identified using exceptions (throw, try and catch)
- Consistency: buttons, icons, colours are kept constant throughout
- Usability- ease of use for user so as to achieve goal with minimal clicks or action and avoid frustration
- Colour Pattern- needs to be kept as a basic but still aesthetically pleasing standard for an enjoyable user experience
- Navigation- the major functions that the system provides are all placed on the homepage so achieving a goal will not be a frustrating task
- Assistance- help prevent the user from doing the wrong thing by making actions of functionalities explicitly clear
- Minimize user's memory load- too much information at once can be a challenge for some users
- Learnability: make the system easy to use by having command shortcuts for example and also make tasks easier to accomplish so even users with little experience with computers can find it easy to learn

1.3 Definitions, Acronyms, and Abbreviations

Abbreviations/Acronyms:

1. DBSM- Database Management System, which is used to store and access user data
2. NF- Non-functional requirements
3. ODD- Object Design System

Definitions:

- Architectural Style- the setting that defines how a system will be developed.
- Constraints- limiting factors to the final output of the product.
- Database- set of stored information such as usernames and passwords.
- Server- the component which manages access to a centralised storage.
- Client- the part responsible for communicating with the server to access information or resources.
- Subsystem- an extract from a larger system.
- Exception Handling- Methods of detection exceptions in a code. Such mechanisms include try, catch and throw.
- Input restriction- limitations to the amount or variety of characters a user can type into an input field
- Off-the-shelf-software- reusing ready made software

1.4 References

The current election system is used generally around the world is on paper election method that requires voters to come to a specific place to vote and besides regional elections for examples in schools, this can be a really big problem.For example last year there was an election in our schools which get %40-50 participation percentage which is very very low. If we want to continue with school example the paperwork needs to be done by the school and candidates and when the election is going through there are a lot of people working in the process to keep the election safe and fast as possible it. After the election is finished there is a calculating votes process and you can’t be sure if it's accurate or not fully. When the election is happening voters need to go to a specific place that is mostly not close to their house or they don’t want to enter the queue for voting. Moreover, current election system is functional but not optimal. In this modern age, the voting process ought to be computerised and this will definitely ensure a greater voters’ turnout and the person in charge of the elections will save precious time and money.

2. Packages

Notification subsystem